

**IN THE UNITED STATES DISTRICT COURT  
FOR THE WESTERN DISTRICT OF TEXAS  
WACO DIVISION**

WSOU INVESTMENTS, LLC d/b/a  
BRAZOS LICENSING AND  
DEVELOPMENT,

Plaintiff,

v.

GOOGLE LLC,

Defendant.

§  
§  
§  
§  
§  
§  
§  
§  
§  
§  
§

CIVIL ACTION NO. 6:20-cv-585

**JURY TRIAL DEMANDED**

**ORIGINAL COMPLAINT FOR PATENT INFRINGEMENT**

Plaintiff WSOU Investments, LLC d/b/a Brazos Licensing and Development (“Brazos” or “Plaintiff”), by and through its attorneys, files this Complaint for Patent Infringement against Google LLC (“Google”) and alleges:

**NATURE OF THE ACTION**

1. This is a civil action for patent infringement arising under the Patent Laws of the United States, 35 U.S.C. §§ 1, *et seq.*, including §§ 271, 281, 284, and 285.

**THE PARTIES**

2. Brazos is a limited liability corporation organized and existing under the laws of Delaware, with its principal place of business at 605 Austin Avenue, Suite 6, Waco, Texas 76701.

3. On information and belief, Google is a Delaware corporation with a physical address at 500 West 2nd Street, Austin, Texas 78701.

**JURISDICTION AND VENUE**

4. This is an action for patent infringement which arises under the Patent Laws of the United States, in particular, 35 U.S.C. §§ 271, 281, 284, and 285.

5. This Court has jurisdiction over the subject matter of this action under 28 U.S.C. §§ 1331 and 1338(a).

6. This Court has specific and general personal jurisdiction over the defendant pursuant to due process and/or the Texas Long Arm Statute, because the defendant has committed acts giving rise to this action within Texas and within this judicial district. The Court's exercise of jurisdiction over the defendant would not offend traditional notions of fair play and substantial justice because the defendant has established minimum contacts with the forum. For example, on information and belief, the defendant has committed acts of infringement in this judicial district, by among other things, selling and offering for sale products that infringe the asserted patent, directly or through intermediaries, as alleged herein.

7. Venue is proper in this Court pursuant to 28 U.S.C. §§ 1391 and 1400(b). Google is registered to do business in Texas. Google has offices in this District, has transacted business in this District, and has committed acts of direct and indirect infringement in this District. Google also has a regular and established place of business in this District, as set forth below.

8. Since 2007, Google has employed "hundreds" of employees in this District in Austin, Texas.<sup>1</sup> As of August 2018, Google had more than 800 employees in Austin.<sup>2</sup> By June of 2019, Google had more than 1,100 employees in Austin.<sup>3</sup> In January 2019, it was reported that Google "signed a lease for an entire 35-story tower that has started construction just east of the

---

<sup>1</sup> According to Gerardo Interiano, Google's public affairs and government relations manager, in a statement. See <http://www.statesman.com/business/google-lease-200-000-square-feet-new-downtown-austin-tower/SANZSa3du8QQ4k8ytOC2rJ/>

<sup>2</sup> See <https://www.statesman.com/news/20190131/source-google-to-occupy-35-story-office-tower-in-downtown-austin>

<sup>3</sup> See <https://www.bizjournals.com/austin/news/2019/06/14/google-confirms-austin-expansion-will-begin-moving.html>

Central Library in downtown Austin.”<sup>4</sup> Google’s 35-story tower in Austin “will have 790,000 square feet of space, enough to potentially house about 5,000 people.”<sup>5</sup>



Source: <https://www.statesman.com/news/20190131/source-google-to-occupy-35-story-office-tower-in-downtown-austin>

9. Articles report that Google’s office in Austin would “would certainly be one of its most expansive offices in North America.”<sup>6</sup>

10. Google has 300,000 square feet of office space in Austin, Texas, at 500 West 2<sup>nd</sup> Street.<sup>7</sup> Google also has offices on North MoPac Expressway,<sup>8</sup> University Park, and Austin’s Children Museum.<sup>9</sup>

---

<sup>4</sup> *Id.*

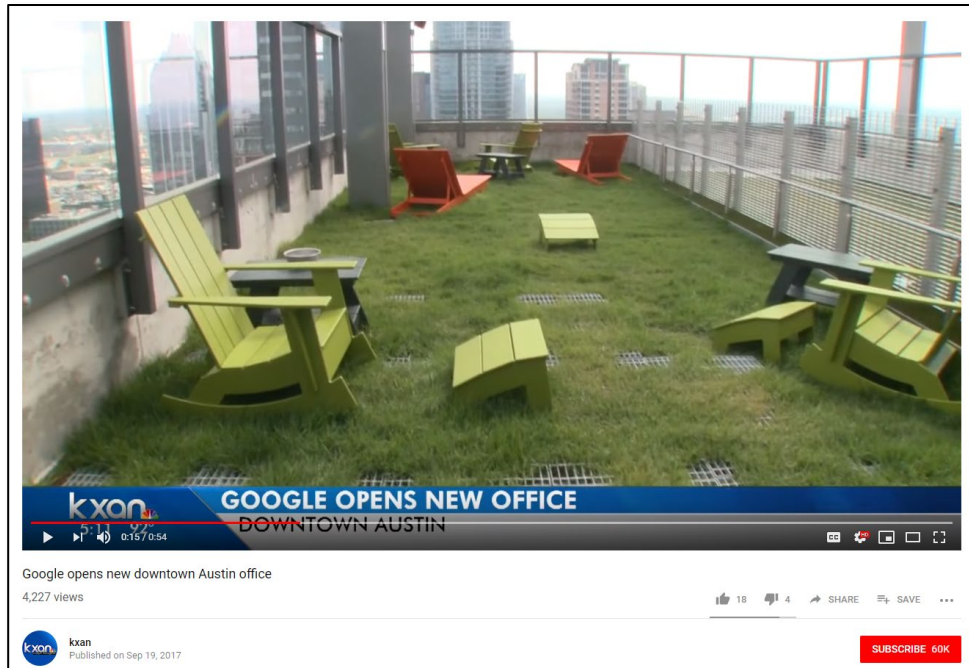
<sup>5</sup> *Id.*

<sup>6</sup> See <https://9to5google.com/2019/01/31/google-signs-lease-austin-campus/>

<sup>7</sup> See <https://www.bizjournals.com/austin/news/2020/02/27/google-to-invest-10b-in-offices-and-data-centers.html>

<sup>8</sup> See <https://www.google.com/intl/en/about/locations/?region=north-america>

<sup>9</sup> See <http://www.statesman.com/business/google-lease-200-000-square-feet-new-downtown-austin-tower/SANZSa3du8QQ4k8ytOC2rJ/>



Source: <https://www.youtube.com/watch?v=RKA1RJYGOYQ>



Source: <https://www.bizjournals.com/austin/news/2019/10/28/inside-austins-coolest-offices.html#g/419929/15>

11. Google has, as of June 2020, fifty (50) job postings for Austin, TX.<sup>10</sup>

12. Google's taxed appraised property values in Travis County (Austin) are approximately \$1 billion.<sup>11</sup> Google's taxed appraised property values in McLennan County (Waco) are approximately \$75,000.<sup>12</sup> Google's taxed appraised property values in Bexar County (San Antonio) are approximately \$50 million.<sup>13</sup> Google's taxed appraised property values in El Paso are approximately \$258,000.<sup>14</sup>

13. Operationally, Google is a multinational technology company that collects, stores, organizes, and distributes data. In addition to its service model for distribution of data (e.g., movies, search results, maps, music, etc.), Google has an expansive regime that gathers data on residents of this District through the hardware devices it sells (e.g., phones, tablets, and home audio devices) and, also, through the operating systems and apps it provides. As an example, Google gathers data when a resident runs its operating systems and apps (e.g., location services).<sup>15</sup> As another example, Google gathers data when a resident interacts with Google's plethora of services such as search, email, and music and movie streaming. See <https://safety.google/privacy/data/> (indicating that Google gathers data from "things you search for," "Videos you watch," "Ads you view or click," "Your location," "Websites you visit," and "Apps, browsers, and devices you use to access Google services"). As yet another example, Google gathers data by listening and recording everything a resident says within proximity of one of its products, such as Google

---

<sup>10</sup>

<https://careers.google.com/jobs/results/?company=Google&company=YouTube&hl=en&jlo=en-US&location=Austin,%20TX,%20USA>

<sup>11</sup> See <http://propaccess.traviscad.org>

<sup>12</sup> See [https://propaccess.trueautomation.com/clientdb/Property.aspx?cid=20&prop\\_id=378970](https://propaccess.trueautomation.com/clientdb/Property.aspx?cid=20&prop_id=378970)

<sup>13</sup> See [https://bexar.acttax.com/act\\_webdev/bexar/showdetail2.jsp?can=000001265355](https://bexar.acttax.com/act_webdev/bexar/showdetail2.jsp?can=000001265355),

<sup>14</sup> See <http://www.epcad.org/Search?Keywords=GOOGLE+INC&Year=2019>

<sup>15</sup> See e.g., "AP Exclusive: Google tracks your movements, like it or not," <https://apnews.com/828aefab64d4411bac257a07c1af0ecb/AP-Exclusive:-Google-tracks-your-movements,-like-it-or-not>

Home.<sup>16</sup> Others have reported that Google gathers “where you’ve been,” “everything you’ve ever searched – and deleted,” “all the apps you use,” “all of your YouTube history,” “which events you attended, and when,” “information you deleted [on your computer],” “your workout routine,” “years’ worth of photos,” and “every email you ever sent.”<sup>17</sup>

14. Google takes these massive amounts of gathered data on residents of this District and monetizes them, for example, through targeted advertising. Some have reported that “creepy” advertisements for items never searched for, but only spoken out loud, appeared. *See e.g.*, <https://www.youtube.com/watch?v=zBnDWSvaQ1I> (conducting test on the term “dog toys” spoken out loud, but never searched; tester claims targeted “dog toy” advertisements only appeared after speaking the phrase out loud).

15. In addition to extensive data gathering of information on residents of this District, Google has a substantial presence in the District directly through the products and services Google provides residents of this District (some of which also gather data).<sup>18</sup> One of Google’s main businesses in this District is delivering information, including digital content such as movies, music, apps, and advertising.

---

<sup>16</sup> *See* <https://www.unilad.co.uk/technology/google-is-listening-to-everything-we-say-and-you-can-hear-it-back/> (“Tech giant and the font of all pub quiz knowledge, Google, can quietly record many of the conversations that people have in close proximity to its products.”).

<sup>17</sup> *See* <https://www.theguardian.com/commentisfree/2018/mar/28/all-the-data-facebook-google-has-on-you-privacy>.

<sup>18</sup> Non-limiting examples include Google Search, Maps, Translate, Chrome Browser, YouTube, YouTube TV, Google Play Music, Chromecast, Google Play Movies and TV, Android Phones, Android Wear, Chromebooks, Android Auto, Gmail, Google Allo, Google Duo, Google+, Google Photos, Google Contacts, Google Calendar, Google Keep, Google Docs, Google Sheets, Google Slides, Google Drive, Google Voice, Google Assistant, Android operating system, Project Fi Wireless phone systems, Google Pixel, Google Home, Google Wifi, Daydream View, Chromecast Ultra.



16. Google describes itself as an “information company.”<sup>19</sup> Its vision is “to provide access to the world’s information in one click,” and its mission is “to organize the world’s information and make it universally accessible and useful.”<sup>20</sup> Making information available to people wherever they are and as quickly as possible is critical to Google’s business.

Google Global Cache (GGC)

17. As Google’s CEO, Sundar Pichai, explains, “We want to make sure that no matter who you are or where you are or how advanced the device you are using—Google works for you.”<sup>21</sup> To meet this goal, Google developed a content delivery network that it calls the Edge Network.

18. One non-limiting example of physical presence in this District is Google’s Edge Network. Google provides web-based services, such as YouTube, YouTube TV, and Google Play, to users throughout the world. These services are in high demand. Google reports that Google Play reaches more than 1 billion Android users and that YouTube serves over 1.8 billion users per month.<sup>22</sup> Studies show that YouTube alone is responsible for approximately 20% of all internet traffic.<sup>23</sup> YouTube TV, which has been described as an “add-on to YouTube” allows Google to essentially become the local TV provider for residents of this District. For example, residents in this District obtain local Waco-Temple-Bryan area channels such as KXXV, ABC (Channel 25); KBTX, CBS (Channel 3) or KWTX, CBS (Channel 10); KCEN NBC (Channel 5); and KCEN,

---

<sup>19</sup> See “This Year’s Founder’s Letter” by Alphabet CEO, Sundar Pichai, <https://blog.google/inside-google/alphabet/this-years-founders-letter/>.

<sup>20</sup> See <http://panmore.com/google-vision-statement-mission-statement>.

<sup>21</sup> See e.g., <http://time.com/4311233/google-ceo-sundar-pichai-letter/>.

<sup>22</sup> See <https://www.theverge.com/2018/5/3/17317274/youtube-1-8-billion-logged-in-monthly-users-brandcast-2018>

<sup>23</sup> See <https://www.sandvine.com/hubfs/downloads/archive/2016-global-internet-phenomena-report-latin-america-and-north-america.pdf> and <http://testinternetspeed.org/blog/half-of-all-internet-traffic-goes-to-netflix-and-youtube/>

Fox (Channel 6).<sup>24</sup> To verify a resident should receive such local channels, Google verifies a location of such resident.

19. Google's Edge Network, itself, has three elements: Core Data Centers, Edge Points of Presence, and Edge Nodes. The Core Data Centers (there are eight in the United States) are used for computation and backend storage. Edge Points of Presence are the middle tier of the Edge Network and connect the Data Centers to the internet. Edge Nodes are the layer of the network closest to users. Popular content, including YouTube TV, YouTube, video advertising, music, mobile apps, and other digital content from the Google Play store, is cached on the Edge Nodes, which Google refers to as Google Global Cache or "GGC."

20. Google Global Cache is recognized as "one of Google's most important pieces of infrastructure,"<sup>25</sup> and Google uses it to conduct the business of providing access to the world's information. GGC servers in the Edge Nodes function as local data warehouses, much like a shoe manufacturer might have warehouses around the country. Instead of requiring people to obtain information from distant Core Data Centers, which would introduce delay, Google stores information in the local GGC servers to provide quick access to the data.

21. Caching and localization are vital for Google's optimization of network resources. Because hosting all content everywhere is inefficient, it makes sense to cache popular content and serve it locally. Doing so brings delivery costs down for Google, network operators, and internet service providers. Storing content locally also allows it to be delivered more quickly, which improves user experience. Serving content from the edge of the network closer to the user improves performance and user happiness. To achieve these benefits, Google has placed Edge Nodes

---

<sup>24</sup> See, e.g. <https://thestreamable.com/markets/waco-temple-bryan-tx>.

<sup>25</sup> See <http://blog.speedchecker.xyz/2015/11/30/demystifying-google-global-cache/>.



throughout the United States, including in this District. Google describes these nodes as the workhorses of video delivery.

22. Just like brick-and-mortar stores, Google's GGC servers independently determine what content to cache based on local requests. The GGC servers in Google's Edge Nodes include software that Google refers to as "µstreamer." µstreamer is responsible for serving video content from YouTube and other Google services, along with other large content such as Google Play applications and Chrome downloads. It operates on a content-delivery platform at the edge of Google's network called "bandaid"; it does not run in the core (except for some internal testing purposes), unlike the majority of the Google services, such as search or gmail.

23. Using µstreamer and bandaid, a GGC server handles requests directly from its clients, predominantly YouTube's video players. When such a request is received, if the content is stored in the node's local cache, the node will serve it to the end user, improving the user experience and saving bandwidth. If cache-eligible content is not already stored on the node, and the content is cache-eligible, the node will retrieve it from Google, serve it to the user, and store it for future requests.

24. µstreamer is largely autonomous, in the sense that almost all decisions related to serving a particular request are made locally, without coordinating with other servers. Like a brick-and-mortar store sells directly to customers from inventory and stocks that inventory based on local customer demand, µstreamer in each GGC node decides—independently from other nodes in Google's Edge Network— whether to serve requested content, whether to cache content, and whether to send requests to other servers.

25. Google's GGC servers are housed in spaces in the District leased by Google. Google's GGC servers are housed in spaces leased by Google from Internet Service Providers

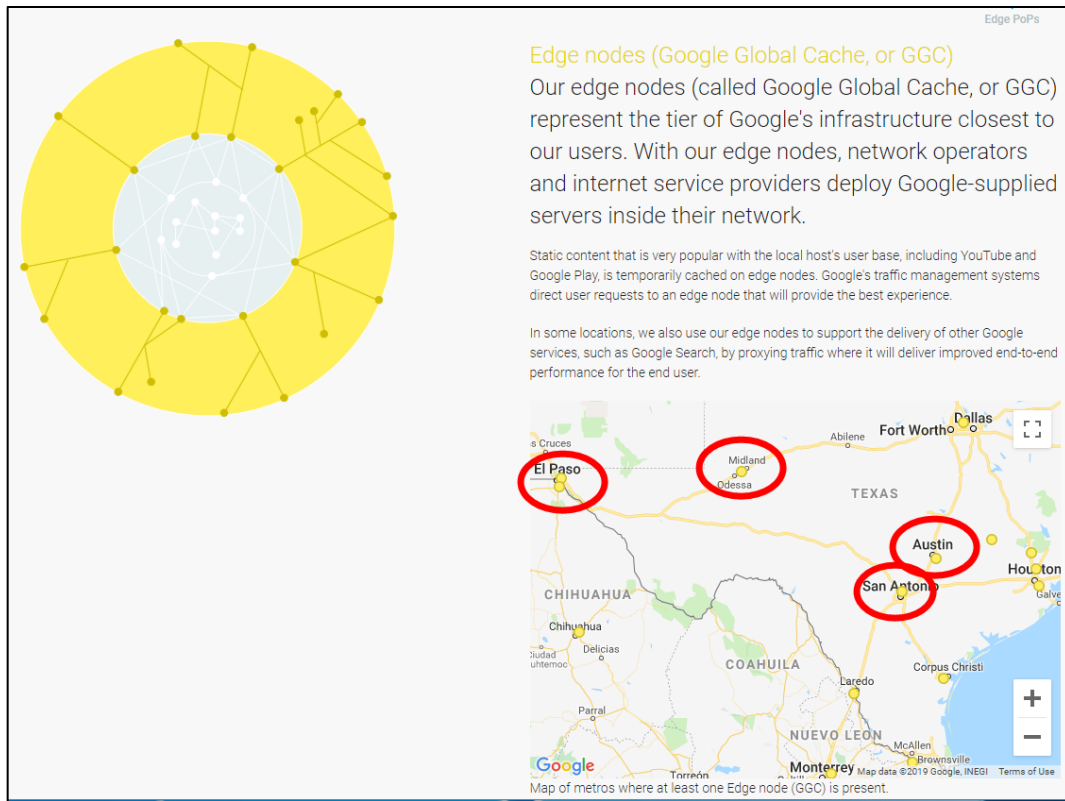
(ISPs) whose networks have substantial traffic to Google and are interested in saving bandwidth. Hosting Google servers allows ISPs to save both bandwidth and costs, as they do not incur the expense of carrying traffic across their peering and/or transit links.

26. When an ISP agrees to host a GGC server, the parties enter into a Global Cache Service Agreement, under which Google provides:

- hardware and software— including GGC servers and software—to be housed in the host’s facilities;
- technical support; service management of the hardware and software; and
- content distribution services, including content caching and video streaming.

In exchange, the host provides, among other things, a physical building, rack space where Google’s computer hardware is mounted, power, and network interfaces. All ownership rights, title, and intellectual property rights in and to the equipment (i.e., the hardware and software provided by Google) remain with Google and/or its licensors.

27. Multiple ISP hosted GGC servers are in this District. Google’s website identifies Midland, El Paso, Austin, and San Antonio as GGC server locations. Each of these cities is located in this District.



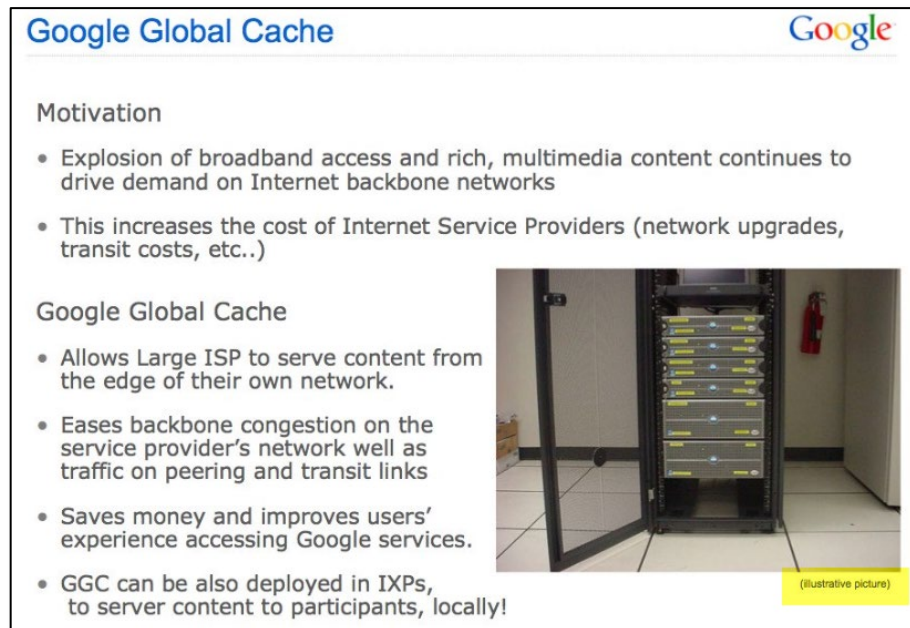
Source: <https://peering.google.com/#/infrastructure>

28. The Office of Telecommunications Services for the University of Texas, for example, is an ISP that hosts two GGC servers in Austin, Texas.<sup>26</sup>
29. Google caches content on the GGC servers located in this District.
30. Google's GGC servers located in this District cache content that includes, among other things: (i) video advertising; (ii) apps; and (iii) digital content from the Google Play store.
31. Google's GGC servers located in this District deliver cached content for the items in the preceding paragraph to residents in this District.
32. Google generates revenue (i) by delivering video advertising, (ii) from apps, and (iii) from digital content in the Google Play store.

<sup>26</sup> See <https://it.utexas.edu/ots-caching-and-peering>

33. Google treats its GGC servers in this District the same as it treats all of its other GGC servers in the United States.

34. The photograph below shows an “illustrative picture” of a Google GGC server.



Source: <https://www.wired.com/2010/03/google-traffic/>

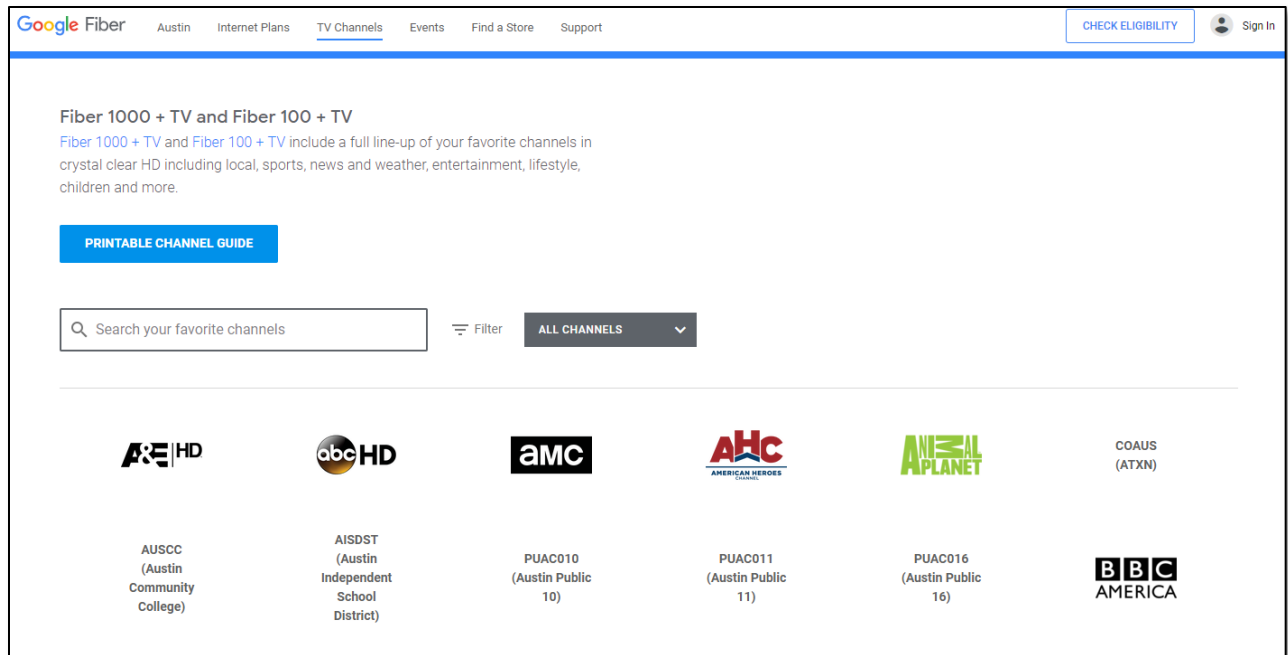
35. Google not only exercises exclusive control over the digital aspects of the GGC, Google, but also exercises exclusive control over the physical server and the physical space within which the server is located and maintained.

#### Google's Communication Services

36. Google provides both data and television services to both San Antonio and Austin.<sup>27</sup>

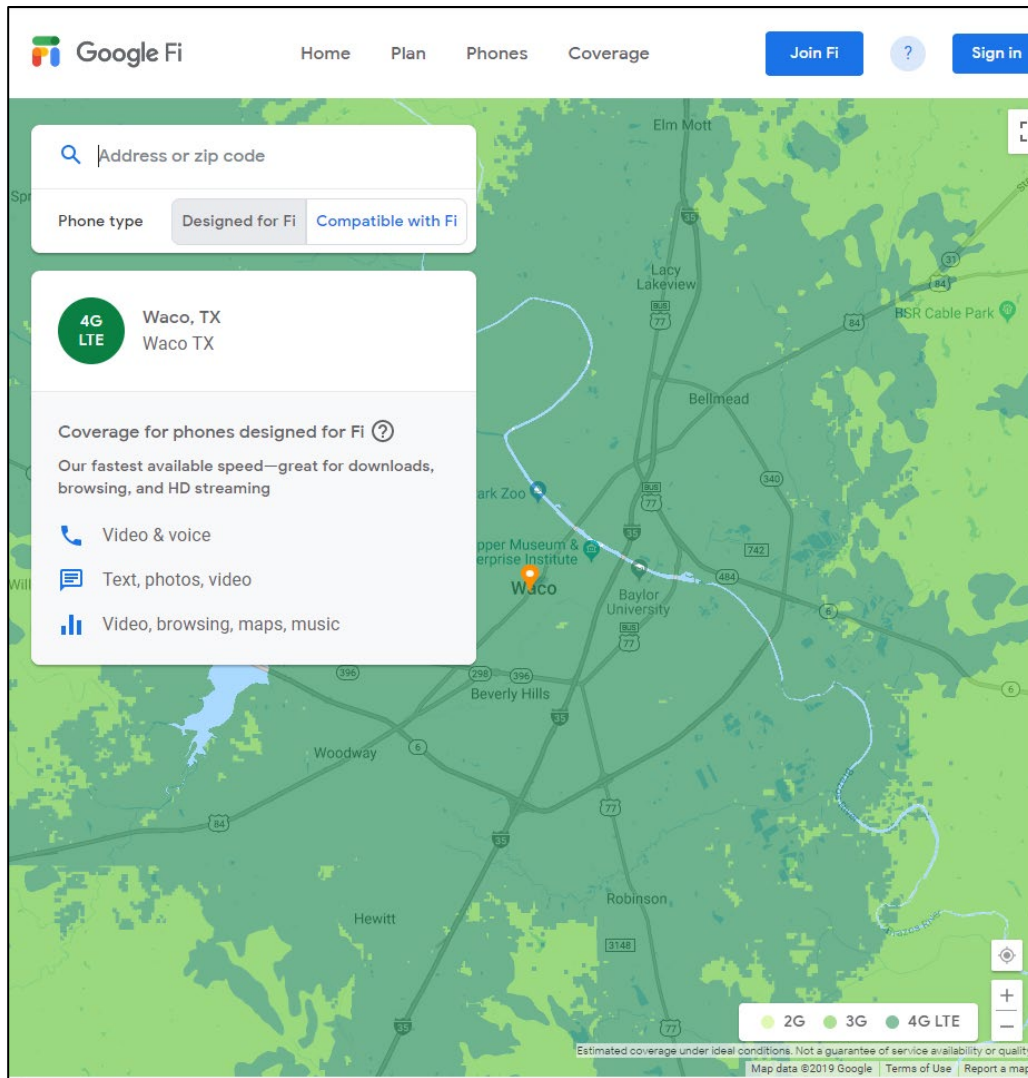
---

<sup>27</sup> <https://fiber.google.com/ourcities/>



### Google's Cell Phone Service (aka Google Fi)

37. Google also provides phone, messaging, and data services in this District from its wireless phone services called Google Fi. Via the Google Fi service, Google provides its customers voice and high-speed data coverage (4G LTE) for cities such as Waco.



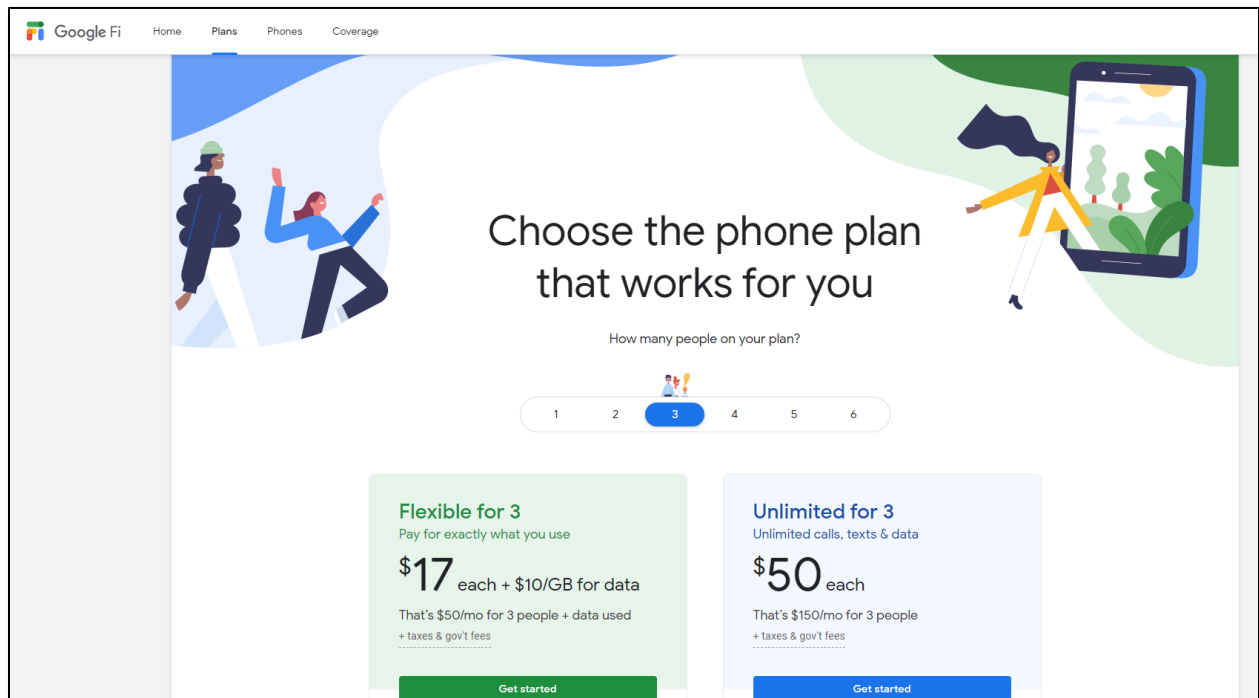
Source: <https://fi.google.com/coverage?q=Waco,%20tx>

38. The cell towers used for Google’s services are fixed geographical locations. They are “regular” and “established” because they operate in a “steady, uniform, orderly, and methodical manner” and are sufficiently permanent. They are “of the defendant” because Google has contractual and/or property rights to use the cell towers to operate its business. Google also ratifies the service locations through its coverage lookup service.

39. With this coverage lookup service, Google advertises its ability to provide cell coverage in this District and its selected cell towers in and near this District to provide the

advertised coverage (e.g., 2G, 3G, or 4GLTE) depending on the location in the District. *See* <https://fi.google.com/coverage?>. Google is not indifferent to the location of its cell towers. It “established” and “ratified” their geographic placement to achieve specific business purposes.

40. Residents of this District also directly contract with and are billed by Google for these services as their telecommunications provider.



Source: <https://fi.google.com/about/plan>

41. Google also determines which cell tower a particular Google Fi customer will use while within the District.

✓ What determines when Project Fi moves me between cellular networks?

When multiple carriers are available, Project Fi will move you to the network that our analysis shows will be fastest in your current location, whether that is 4G LTE, 3G, or 2G. We're constantly learning and improving, to account for factors such as newly-built towers or newly-available radio frequencies. And if your current network is providing weak or no coverage, we'll adjust in real time to find you a stronger connection.



Source: <https://fi.google.com/about/faq/#network-and-coverage-4>

**COUNT ONE - INFRINGEMENT OF U.S. PATENT NO. 8,737,961**

42. Brazos re-alleges and incorporates by reference the preceding paragraphs of this Complaint.

43. On May 27, 2014, the United States Patent and Trademark Office duly and legally issued U.S. Patent No. 8,737,961 (“the ‘961 Patent”), entitled “Method and apparatus for incrementally determining location context.” A true and correct copy of the ‘961 Patent is attached as Exhibit A to this Complaint.

44. Brazos is the owner of all rights, title, and interest in and to the ‘961 Patent, including the right to assert all causes of action arising under the ‘961 Patent and the right to any remedies for the infringement of the ‘961 Patent.

45. Google makes, uses, sells, offers for sale, imports, and/or distributes in the United States, including within this judicial district, products such as, but not limited to, systems that incrementally determine the location context of a mobile device (collectively, the “Accused Products”).

46. The Accused Products include, but are not limited to, Google’s Awareness API.

47. The Awareness API is an API providing multiple context and location signals to mobile applications. The Awareness API comprises five location and context signals: time, location, activity, beacons, and headphones. The Awareness API includes a Fence API and a Snapshot API. The Fence API allows an app to register the current situation of a user and conveys

notification of the met combined context conditions. The Snapshot API allows an app to send requests for information about the current context of a user.

48. The Awareness API unifies location and context signals in a single API. The Awareness API enables developers to create context-based features with minimal impact on system resources. An application combines optimally processed context signals and lets the API manage system resources so that the app does not have to do the same. The API supports different signals such as time, location, places, etc.

## Anticipate and react

The Awareness API unifies 7 location and context signals in a single API, enabling you to create powerful context-based features with minimal impact on system resources. Combine optimally processed context signals in new ways that were not previously possible, while letting the API manage system resources so your app doesn't have to.



### Many signals, one API

Native support for combining and working with 7 signals including time, location, places, beacons, headphones, activity and weather.



### High quality data

Signals from multiple sources are intelligently processed, and combined for maximum accuracy and efficiency.



### Smart battery savings

Power consumption and memory usage are automatically optimized to maximize battery life and memory capacity on your users' devices.

Source: <https://developers.google.com/awareness>.

## What's the Awareness API?

With the Google Awareness API, you can enable your app to intelligently react to the user's current situation. The Awareness API exposes five different [context types](#), which include user activity, and nearby beacons. These types enable your app to refine the user experience in new ways that weren't possible before. Your app can combine these context signals to make inferences about the user's current situation, and use this information to provide customized experiences, such as a playlist suggestion when the user plugs in their headphones and starts to jog.

Source: <https://developers.google.com/awareness/overview>

49. The Awareness API offers contexts for different user situations. The contextual data includes sensor-derived signal data such as location (e.g. latitude and longitude), place type, like a park or coffee shop, and activity, such as a walk or drive.

### Context types

Context is at the heart of the Awareness API. Contextual data includes sensor-derived data such as location (latitude and longitude), place type, like a park or coffee shop, and activity, such as a walk or drive. These basic types, or signals, can be combined to extrapolate the user's situation in more specific detail. Expand the following notice to see which contextual signals have been deprecated.

Source: <https://developers.google.com/awareness/overview>

50. The Awareness API uses Fence API and Snapshot API to get context signals to determine the user's current situation.

51. The Fence API allows an app to register the current situation of a user and conveys notifications when a combination of context conditions are met. The Snapshot API allows an app to request information about the current context of a user.

## Fences and snapshots

The Awareness API consists of two distinct APIs that your app can use to get context signals in order to determine the user's current situation:

- **Fence API:** This API lets your app react to the user's current situation, and provides notification when a combination of context conditions are met. For example, whenever the user takes a walk and their headphones are plugged in. Once a fence is registered, the Fence API can send callbacks to your app even when it's not running.
- **Snapshot API:** This API lets your app request information about the user's current context, such as the user's current location and the current weather conditions.

Source: <https://developers.google.com/awareness/overview>

Context type	Example
Time	Current local time
Location	Latitude and longitude
Activity	Detected user activity, like walking, running, or biking
Beacons	Nearby beacons that match the specified namespace
Headphones	Status of whether headphones are plugged in, or not

Source: <https://developers.google.com/awareness/overview>

52. The Awareness API gets contextual signal data from different sensors and interference sources (or, set of one or more distinct signal sources) continuously over different time periods. For example, the Snapshot API captures sensors data (e.g. motion sensing data) to

determine activities and locations. Signal data such as Location captured from different devices sources (e.g. position sensors) include different components like latitude, longitude, timestamps, etc. Thus, the mobile device receives signal data from different sources at different times.

You can use the [Snapshot API](#) to get information about the user's current environment. With the Snapshot API, you can access a variety of [context signals](#):

- Detected user activity, such as when they walk or drive.
- Nearby beacons that you've registered.
- Headphone state, plugged in or not.
- Location, which includes latitude and longitude.

The system caches these values so that they can be returned quickly. If there's no data, sensing and inference are performed to return fresh state values. The Awareness API returns the existing data type for context types that have a public API.

Source: <https://developers.google.com/awareness/android-api/snapshot-api-overview>

## Get location

You can get the user's current location (latitude-longitude) with a call to `getLocation()`, which returns a `LocationResponse`. You can then call `LocationResponse.getLocation()` to get a `Location` [↗](#) with the current location data.

The `getLocation()` method requires the `android.permission.ACCESS_FINE_LOCATION` permission. Add this permission to `AndroidManifest.xml`.

Source: <https://developers.google.com/awareness/android-api/snapshot-get-data>

# Location

Added in API level 1

Kotlin | Java

```
public class Location
extends Object implements Parcelable

java.lang.Object
└─ android.location.Location
```

A data class representing a geographic location.

A location can consist of a latitude, longitude, timestamp, and other information such as bearing, altitude and velocity.

All locations generated by the `LocationManager` are guaranteed to have a valid latitude, longitude, and timestamp (both UTC time and elapsed real-time since boot), all other parameters are optional.

Source: <https://developer.android.com/reference/android/location/Location.html>

53. The Awareness API uses the Fence API and the Snapshot API to allow apps to get context signals to determine a user's current situation. For example, the Snapshot API requests information about the user's current context signal such as the location for different times. The Awareness API includes the Fence API to let an application react to the user's current situation and provide notification when a combination of context conditions are met.

## Fences and snapshots

The Awareness API consists of two distinct APIs that your app can use to get context signals in order to determine the user's current situation:

- **Fence API:** This API lets your app react to the user's current situation, and provides notification when a combination of context conditions are met. For example, whenever the user takes a walk and their headphones are plugged in. Once a fence is registered, the Fence API can send callbacks to your app even when it's not running.
- **Snapshot API:** This API lets your app request information about the user's current context, such as the user's current location and the current weather conditions.

Source: <https://developers.google.com/awareness/overview>.


You can use the [Snapshot API](#) to get information about the user's current environment. With the Snapshot API, you can access a variety of [context signals](#):

- Detected user activity, such as when they walk or drive.
- Nearby beacons that you've registered.
- Headphone state, plugged in or not.
- Location, which includes latitude and longitude.

The system caches these values so that they can be returned quickly. If there's no data, sensing and inference are performed to return fresh state values. The Awareness API returns the existing data type for context types that have a public API.

Source: <https://developers.google.com/awareness/android-api/snapshot-api-overview>

54. The Fence API uses geofencing which helps in determining a user's proximity to locations of interest. Geofencing enables developers to add a radius to adjust the proximity of a specific location and the duration in that specific location, hence determining whether a user's mobile device is moving outside a specified area at a current time based on signal data. Using the Fence API, an app receives callbacks when a user enters or leaves the geofence region.

In the Awareness API, the concept of *fences* is taken from [geofencing](#) , in which a geographic region, or *geofence*, is defined, and an app receives callbacks when a user enters or leaves the geofence region. The Fence API expands on the concept of geofencing to include many other context conditions in addition to geographical proximity. An app receives callbacks whenever the context state transitions. For example, if your app defines a fence for headphones, it gets callbacks when the headphones are plugged in and when they're unplugged.

Source: <https://developers.google.com/awareness/android-api/fence-api-overview>



## Create and monitor geofences

Geofencing combines awareness of the user's current location with awareness of the user's proximity to locations that may be of interest. To mark a location of interest, you specify its latitude and longitude. To adjust the proximity for the location, you add a radius. The latitude, longitude, and radius define a geofence, creating a circular area, or fence, around the location of interest.

You can have multiple active geofences, with a limit of 100 per app, per device user. For each geofence, you can ask Location Services to send you entrance and exit events, or you can specify a duration within the geofence area to wait, or *dwell*, before triggering an event. You can limit the duration of any geofence by specifying an expiration duration in milliseconds. After the geofence expires, Location Services automatically removes it.

Source: <https://developer.android.com/training/location/geofencing>

First, use `Geofence.Builder` to create a geofence, setting the desired radius, duration, and transition types for the geofence. For example, to populate a list object:

Source: <https://developer.android.com/training/location/geofencing>

55. The Awareness API determines the situation for a user device based on the signal data received from different sources such as location data, activity data, etc. The Fence API determines whether a user enters or leaves a region using geofencing. This allows the Fence API to determine a primary set of stationary states of a device based on continuous tracking of sensing signal data.

In the Awareness API, the concept of *fences* is taken from [geofencing](#) [\[1\]](#), in which a geographic region, or *geofence*, is defined, and an app receives callbacks when a user enters or leaves the geofence region. The Fence API expands on the concept of geofencing to include many other context conditions in addition to geographical proximity. An app receives callbacks whenever the context state transitions. For example, if your app defines a fence for headphones, it gets callbacks when the headphones are plugged in and when they're unplugged.

Source: <https://developers.google.com/awareness/android-api/fence-api-overview>

## Create and monitor geofences

Geofencing combines awareness of the user's current location with awareness of the user's proximity to locations that may be of interest. To mark a location of interest, you specify its latitude and longitude. To adjust the proximity for the location, you add a radius. The latitude, longitude, and radius define a geofence, creating a circular area, or fence, around the location of interest.

You can have multiple active geofences, with a limit of 100 per app, per device user. For each geofence, you can ask Location Services to send you entrance and exit events, or you can specify a duration within the geofence area to wait, or *dwell*, before triggering an event. You can limit the duration of any geofence by specifying an expiration duration in milliseconds. After the geofence expires, Location Services automatically removes it.

Source: <https://developer.android.com/training/location/geofencing>

56. The Fence API allows combining multiple context signals to create fences with Boolean operators. This allows an application to receive callbacks when the fence conditions are met. The user can use combination fences as well. This allows the association of a context signal (e.g. stationary state i.e. within a geofencing region) with another distinct context signal (e.g. activity data – walking, running, etc.).

The Fence API lets you combine multiple **context signals** to create fences with **AND**, **OR**, and **NOT** boolean operators. Your app then receives callbacks whenever the fence conditions are met. Some examples of possible fences include the following:

- User plugs in headphones and starts to walk.
- User enters a 100-meter geofence before 5 PM on a weekday.
- User enters range of a specific BLE beacon.

Source: <https://developers.google.com/awareness/android-api/fence-api-overview>

Once you've defined a fence, you must do the following:

- Call **updateFences** to register the fence to receive callbacks.
- Define a callback that can be invoked when the fence state changes.

Source: <https://developers.google.com/awareness/android-api/fence-api-overview>

## Create a fence

A fence defines one or more context conditions to which your app can react. When a fence's state changes, your app receives a callback.

There are two types of fences: primitive fences, which represent the basic set of context signals, and combination fences, which combine multiple primitive fences with the use of boolean operators. All fences are instances of `AwarenessFence`.

Source: <https://developers.google.com/awareness/android-api/fence-create>

57. Geofencing allows the determination of a stationary state based on fences. A stationary state can be combined with an incremented count for another signal source such as motion data for activity (e.g. walking) using one or more sensors such as motion sensors, position sensors, etc. For example, when a user's stationary state (i.e. within a geofencing region) is determined, the application can incrementally count data for other context sets (e.g. step counting, distance covered, etc.).

Nested trees of `AND`, `OR` and `NOT` are valid, so any boolean combination of fences is possible. The following example shows a fence that's triggered when a user moves more than 100 meters from the current location, or over an hour has elapsed since the current time.

```
double currentLocationLat; // current location latitude
double currentLocationLng; // current location longitude
long nowMillis = System.currentTimeMillis();
long oneHourMillis = 1L * 60L * 60L * 1000L;

AwarenessFence orExample = AwarenessFence.or(
    AwarenessFence.not(LocationFence.in(
        currentLocationLat,
        currentLocationLng,
        100.0,
        100.0,
        0L)),
    TimeFence.inInterval(nowMillis + oneHourMillis, Long.MAX_VALUE));
```

Source: <https://developers.google.com/awareness/android-api/fence-create>

## Motion sensors

The Android platform provides several sensors that let you monitor the motion of a device.

The sensors' possible architectures vary by sensor type:

- The gravity, linear acceleration, rotation vector, significant motion, step counter, and step detector sensors are either hardware-based or software-based.
- The accelerometer and gyroscope sensors are always hardware-based.

Source: [https://developer.android.com/guide/topics/sensors/sensors\\_motion](https://developer.android.com/guide/topics/sensors/sensors_motion)

## Use the step counter sensor

The step counter sensor provides the number of steps taken by the user since the last reboot while the sensor was activated. The step counter has more latency (up to 10 seconds) but more accuracy than the step detector sensor.

Source: [https://developer.android.com/guide/topics/sensors/sensors\\_motion](https://developer.android.com/guide/topics/sensors/sensors_motion)

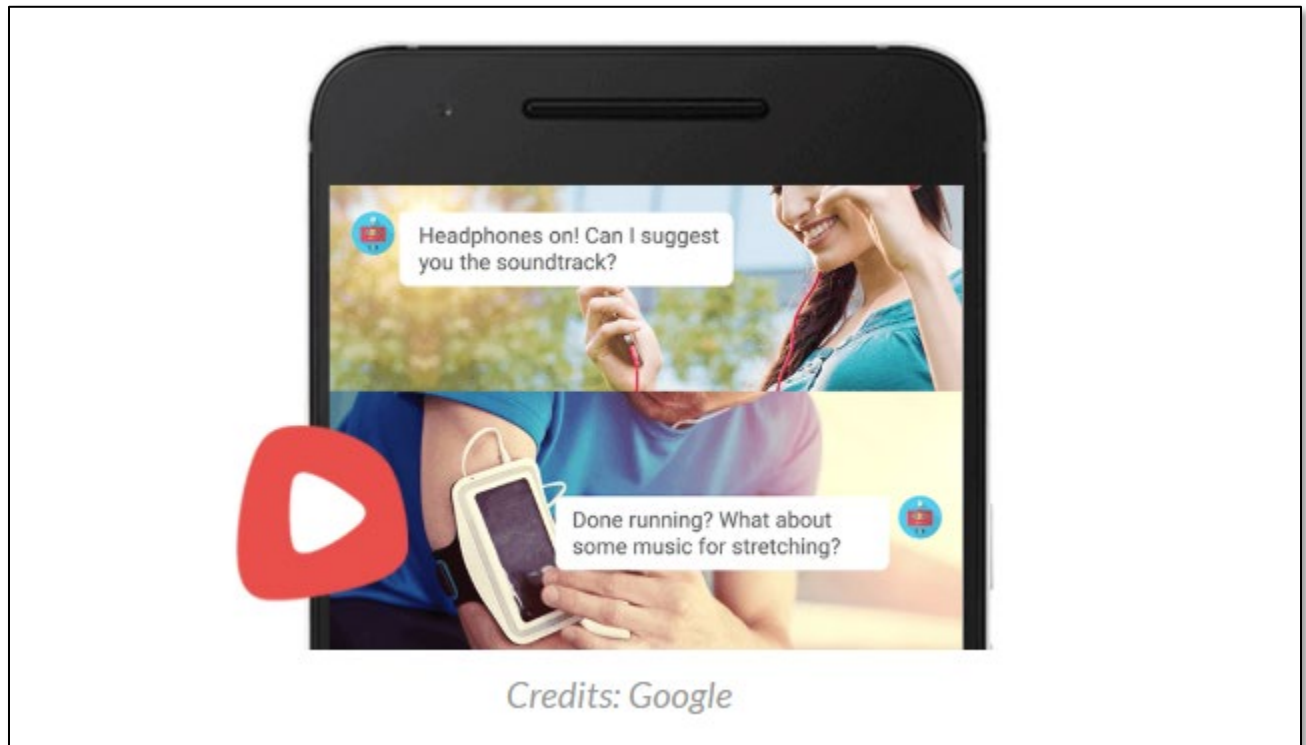
58. The Fence API and the Snapshot API get signal data for different contexts of a mobile application. The Fence API allows the determination of stationary state based on geofencing. The application receives callbacks whenever the context state transitions. These callbacks allow delivery of a service when a context signal is determined (e.g. stationary state).

## Fence API overview

In the Awareness API, the concept of *fences* is taken from [geofencing](#), in which a geographic region, or *geofence*, is defined, and an app receives callbacks when a user enters or leaves the geofence region. The Fence API expands on the concept of geofencing to include many other context conditions in addition to geographical proximity. An app receives callbacks whenever the context state transitions. For example, if your app defines a fence for headphones, it gets callbacks when the headphones are plugged in and when they're unplugged.

Source: <https://developers.google.com/awareness/android-api/fence-api-overview>

59. As an example, when a user connects headphones with the mobile device, the application can provide a request for suggesting soundtracks. An application can suggest other services based on the determination of a stationary state e.g. news.



Source: <https://blog.iamsuleiman.com/introduction-googles-awareness-api/>

We can also use the Snapshot API to detect the current activity that the user is currently engaging in. Some simple use cases could be found in:

- A fitness app where we wish to detect when the user has stopped running so we can automatically pause their activity 🏃
- A news app where we wish to detect when the user is stationary and provide them with something to read during this time 📰

Source: <https://medium.com/exploring-android/exploring-the-new-google-awareness-api-bf45f8060bba>

60. In view of preceding paragraphs, each and every element of at least claim 1 of the ‘961 Patent is found in the Accused Products.

61. Google continues to directly infringe at least one claim of the ‘961 Patent, literally or under the doctrine of equivalents, by making, using, selling, offering for sale, importing, and/or distributing the Accused Products in the United States, including within this judicial district, without the authority of Brazos.

62. Google has received notice and actual or constructive knowledge of the ‘961 Patent since at least the date of service of this Complaint.

63. Since at least the date of service of this Complaint, through its actions, Google has actively induced product makers, distributors, retailers, and/or end users of the Accused Products to infringe the ‘961 Patent throughout the United States, including within this judicial district, by, among other things, advertising and promoting the use of the Accused Products in various websites, including providing and disseminating product descriptions, operating manuals, and other instructions on how to implement and configure the Accused Products. Examples of such advertising, promoting, and/or instructing include the documents at:

- <https://developers.google.com/awareness>

- <https://developers.google.com/awareness/overview>
- <https://developers.google.com/awareness/android-api/snapshot-api-overview>
- <https://developers.google.com/awareness/android-api/snapshot-get-data>
- <https://developer.android.com/training/location/geofencing>
- <https://blog.iamsuleiman.com/introduction-googles-awareness-api/>
- <https://developer.android.com/reference/android/location/Location.html>
- <https://medium.com/exploring-android/exploring-the-new-google-awareness-api-bf45f8060bba>
- <https://developers.google.com/awareness/android-api/fence-api-overview>
- <https://developers.google.com/awareness/android-api/fence-create>
- [https://developer.android.com/guide/topics/sensors/sensors\\_motion](https://developer.android.com/guide/topics/sensors/sensors_motion)

64. Since at least the date of service of this Complaint, through its actions, Google has contributed to the infringement of the ‘961 Patent by having others sell, offer for sale, or use the Accused Products throughout the United States, including within this judicial district, with knowledge that the Accused Products infringe the ‘961 Patent. The Accused Products are especially made or adapted for infringing the ‘961 Patent and have no substantial non-infringing use. For example, in view of the preceding paragraphs, the Accused Products contain functionality which is material to at least one claim of the ‘961 Patent.

#### **JURY DEMAND**

Brazos hereby demands a jury on all issues so triable.

#### **REQUEST FOR RELIEF**

WHEREFORE, Brazos respectfully requests that the Court:

(A) Enter judgment that Google infringes one or more claims of the ‘961 Patent literally and/or under the doctrine of equivalents;



(B) Enter judgment that Google has induced infringement and continue to induce infringement of one or more claims of the '961 Patent;

(C) Enter judgment that Google has contributed to and continue to contribute to the infringement of one or more claims of the '961 Patent;

(D) Award Brazos damages, to be paid by Google in an amount adequate to compensate Brazos for such damages, together with pre-judgment and post-judgment interest for the infringement by Google of the '961 Patent through the date such judgment is entered in accordance with 35 U.S.C. § 284, and increase such award by up to three times the amount found or assessed in accordance with 35 U.S.C. § 284;

(E) Declare this case exceptional pursuant to 35 U.S.C. § 285; and

(F) Award Brazos its costs, disbursements, attorneys' fees, and such further and additional relief as is deemed appropriate by this Court.

Dated: June 29, 2020

Respectfully submitted,

/s/ James L. Etheridge

James L. Etheridge

Texas State Bar No. 24059147

Ryan S. Loveless

Texas State Bar No. 24036997

Travis L. Richins

Texas State Bar No. 24061296

ETHERIDGE LAW GROUP, PLLC

2600 E. Southlake Blvd., Suite 120 / 324

Southlake, Texas 76092

Telephone: (817) 470-7249

Facsimile: (817) 887-5950

[Jim@EtheridgeLaw.com](mailto:Jim@EtheridgeLaw.com)

[Ryan@EtheridgeLaw.com](mailto:Ryan@EtheridgeLaw.com)

[Travis@EtheridgeLaw.com](mailto:Travis@EtheridgeLaw.com)

**COUNSEL FOR PLAINTIFF**